

FLASH CARDS

Prompt engineering

Terminology and quiz cards for the lesson on writing clear instructions for language models.

What is in this set

40 cards: **24 term cards** (term on the front, definition on the back) and **16 quiz cards** (question on the front, answer on the back). Bracketed numbers on the backs, like [1], match the reference list on the last page and in the instructor guide.

How to print

1. Print double-sided on US Letter at 100% scale (no "fit to page").
2. Choose **flip on the long edge**. The backs are arranged so each answer lands behind its card.
3. Cut along the dashed lines. Each card is 3.75 by 2.5 inches.

How to use them

- **Self-test:** read the front, answer aloud, flip to check.
- **Pairs:** one reads the front, the other answers. Swap every five cards.
- **Boxes:** keep a "missed" pile and a "got it" pile. Re-test the missed pile after a break, then again the next day.
- **Warm-up:** deal a few quiz cards at the start of the next session.

Facts checked October 27, 2026. Tools and models change quickly, so confirm details against the current documentation. These materials are dedicated to the public domain under CC0 1.0 and shared for the public good. The PrivacySafe and PrivacySafe Foundation names and logos are trademarks and are not included in that dedication.

TERM

1 / 40

Prompt

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

2 / 40

System message

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

3 / 40

User message

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

4 / 40

Token

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

5 / 40

Context window

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

6 / 40

Temperature

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

7 / 40

Zero-shot prompt

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

8 / 40

Few-shot prompt

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

2 / 40

Instructions from whoever built the app that set role, tone and rules. Many APIs give it more authority than the user message.

[\[1\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

1 / 40

The text you give a model: instructions, context, examples and the data to work on.

[\[1\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

4 / 40

A chunk of text, often part of a word, that the model reads and writes. English averages roughly three quarters of a word per token.

General

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

3 / 40

The request from the person using the model.

[\[1\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

6 / 40

A setting that controls randomness in sampling. Lower is more repeatable. Higher is more varied.

[\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

5 / 40

The most tokens a model can consider at once, counting the prompt and the reply. Anything outside it is not seen.

General

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

8 / 40

A prompt that includes a few examples so the model can copy the pattern.

[\[3\]](#) [\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

7 / 40

Instructions with no examples.

[\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

9 / 40

Chain of thought

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

10 / 40

Zero-shot chain of thought

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

11 / 40

Self-consistency

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

12 / 40

Prompt chaining

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

13 / 40

Role or audience

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

14 / 40

Constraint

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

15 / 40

Output format

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

16 / 40

Delimiters

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

10 / 40

Adding a trigger phrase such as "Let's think step by step" without giving examples.

[5]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

9 / 40

Asking for, or showing, intermediate reasoning steps before the final answer.

[4]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

12 / 40

Splitting a big job into steps, where the output of one prompt feeds the next.

[12]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

11 / 40

Sampling several reasoning paths and taking the most common final answer.

[6]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

14 / 40

A measurable limit on the answer: length, format, tone, or what to skip.

[2]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

13 / 40

Telling the model who is answering or who the answer is for, to set tone and level.

[1]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

16 / 40

Markers such as tags or triple quotes that separate your instructions from pasted data.

[8]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

15 / 40

The shape of the reply: bullets, a table, JSON with named keys, or one label.

[2]

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

17 / 40

Structured output

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

18 / 40

Prompt template

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

19 / 40

Test set

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

20 / 40

Hallucination

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

21 / 40

Prompt injection

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

22 / 40

Indirect prompt injection

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

23 / 40

Least privilege

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

24 / 40

Prompt leakage

Say it in your own words, then flip.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

18 / 40

A reusable prompt with variables for the parts that change each time.

[\[12\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

17 / 40

A reply in a machine-readable format, such as JSON, that a program can validate.

[\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

20 / 40

Fluent output that is false or unsupported. Also called confabulation.

[\[14\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

19 / 40

A small collection of real and tricky inputs, with what a good answer looks like, used to check a prompt.

[\[12\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

22 / 40

Hidden instructions in content the model reads, such as a web page, file or email.

[\[8\]](#) [\[10\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

21 / 40

Input that makes a model follow an attacker's instructions instead of yours.

[\[8\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

24 / 40

Revealing the system prompt. It is why you should never put secrets in one.

[\[8\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

TERM

23 / 40

Giving the model and its tools only the access they need, to limit damage if a prompt is subverted.

[\[8\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

25 / 40

Name the six parts of a good prompt.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

26 / 40

Why can the same prompt give different answers?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

27 / 40

What does a model see when you send a prompt?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

28 / 40

Few-shot versus zero-shot: what is the difference?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

29 / 40

When might step-by-step prompting help?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

30 / 40

Why wrap pasted text in tags?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

31 / 40

What is wrong with "Write something good"?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

32 / 40

How would you fix "Make it short but detailed"?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

26 / 40

Sampling adds randomness, controlled by temperature. Models and versions also differ.

[\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

25 / 40

Task, context, role or audience, constraints, output format, and examples.

[\[1\]](#) [\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

28 / 40

Few-shot includes a few examples. Zero-shot gives instructions only.

[\[3\]](#) [\[2\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

27 / 40

Only the text in its context window: system message, your prompt and any data. No files or memory unless you supply them.

[\[1\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

30 / 40

It separates your instructions from the data and helps the model. It slows simple injection but is not a guarantee.

[\[8\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

29 / 40

On multi-step problems such as math and logic. Models trained to reason may already do it.

[\[4\]](#) [\[5\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

32 / 40

It conflicts with itself. Pick one, or give numbers, such as a 2-sentence summary then 3 detail bullets.

[General](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

31 / 40

It is vague: no audience, no format, no limits, and no definition of good.

[General](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

33 / 40

What is prompt injection?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

34 / 40

Direct versus indirect injection?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

35 / 40

Is a prompt a security boundary?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

36 / 40

Why keep a test set?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

37 / 40

Why change one thing at a time?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

38 / 40

Why re-test when you switch models?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

39 / 40

Name two things not to paste into a hosted AI tool.

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

40 / 40

Where should key details go in a long prompt?

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

34 / 40

Direct comes from the user. Indirect hides in content the model reads, such as a web page, file or email.

[\[8\]](#) [\[10\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

33 / 40

Input that makes the model follow an attacker's instructions instead of yours.

[\[8\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

36 / 40

To see whether a change really helped and to catch new failures. Include edge cases.

[\[12\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

35 / 40

No. Use least privilege, validate output, and keep a human in the loop. Design as if the model might ignore it.

[\[8\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

38 / 40

Models differ, and small wording or formatting changes can shift results.

[\[11\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

37 / 40

Otherwise you cannot tell which change caused the result.

[General](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

40 / 40

Near the start or the end. Details buried in the middle can be missed.

[\[7\]](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

QUIZ

39 / 40

Passwords, API keys, secrets, and personal data. Redact them, or use a local model.

[General](#)

Prompt engineering

PrivacySafe Foundation | CC0 1.0

Reference list

Links were checked on October 27, 2026.

#	SOURCE	WHAT IT SUPPORTS
[1]	OpenAI API documentation, Prompt engineering https://developers.openai.com/api/docs/guides/prompt-engineering	Message roles with different levels of authority; role and tone in the system message, task details and examples in user messages; few-shot structure.
[2]	Google, Prompt design strategies (Gemini API) https://ai.google.dev/gemini-api/docs/prompting-strategies	Clear instructions, constraints, response format, and the recommendation to include few-shot examples.
[3]	Brown et al., Language Models are Few-Shot Learners (2020) https://arxiv.org/abs/2005.14165	In-context learning from a few examples in the prompt.
[4]	Wei et al., Chain-of-Thought Prompting Elicits Reasoning in Large Language Models (2022) https://arxiv.org/abs/2201.11903	Examples that include intermediate reasoning steps improve multi-step reasoning in sufficiently large models.
[5]	Kojima et al., Large Language Models are Zero-Shot Reasoners (2022) https://arxiv.org/abs/2205.11916	A trigger phrase such as "Let's think step by step" can elicit reasoning without examples.
[6]	Wang et al., Self-Consistency Improves Chain of Thought Reasoning (2022) https://arxiv.org/abs/2203.11171	Sampling several reasoning paths and taking the most consistent answer.
[7]	Liu et al., Lost in the Middle: How Language Models Use Long Contexts (2023) https://arxiv.org/abs/2307.03172	Models can use information near the start and end of a long context better than information in the middle.
[8]	OWASP GenAI Security Project, LLM01:2025 Prompt Injection https://genai.owasp.org/llmrisk/llm01-prompt-injection/	Definitions of direct and indirect prompt injection, example attacks, and mitigations such as least privilege and human review.
[9]	Perez and Ribeiro, Ignore Previous Prompt: Attack Techniques for Language Models (2022) https://arxiv.org/abs/2211.09527	Early research on goal hijacking and prompt leaking attacks.
[10]	Greshake et al., Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection (2023) https://arxiv.org/abs/2302.12173	Indirect prompt injection through content a model retrieves or reads.
[11]	Sclar et al., Quantifying Language Models' Sensitivity to Spurious Features in Prompt Design (2023) https://arxiv.org/abs/2310.11324	Small changes in prompt formatting can change results substantially, so re-test when wording or models change.
[12]	DAIR.AI, Prompt Engineering Guide https://www.promptingguide.ai/	A free, community-maintained overview of techniques such as prompt chaining and templates.
[13]	Ollama documentation, FAQ https://docs.ollama.com/faq	Running a local server on 127.0.0.1 and allowing extra web origins with OLLAMA_ORIGINS.
[14]	NIST, AI Risk Management Framework https://www.nist.gov/itl/ai-risk-management-framework	A framework for managing AI risks, including those of generative AI.
[15]	Creative Commons, CC0 1.0 Universal https://creativecommons.org/publicdomain/zero/1.0/	The public-domain dedication that covers these lesson materials.